

### Remarks

Entrance of this amendment and allowance of the remaining claims are respectfully requested. Claims 1, 3, 4, 6, & 8-10 remain pending.

By this paper, independent claim 1 is amended to more particularly point out and distinctly claim certain novel aspects of the present invention. Specifically, claim 1 is amended to specify that *only the update control code is copied* from the currently executing, first software module to the memory space outside the memory location at which the first software module is currently executing. Further, the claim is amended to specify that the executing of the update control code during the replacing is to monitor replacing of the first software module with the second software module, and *that responsive to the replacing, and without resetting or restarting the system, Applicant's non-disruptive method includes branching from the executing of the updated control code to executing code of the second software module at an entry point thereof specified by the update control code, thereby beginning execution of the second software module.* Still further, Applicant specifies that the first software module and the second software module are each a single statically linked module. Support for amended claim 1 can be found throughout the application as filed. For example, reference previously pending dependent claims 4, 5 & 7, as well as FIGS. 3, 4 & 5, and the supporting discussion thereof in the specification, beginning at paragraph [0026]. No new matter is added to the application by any amendment presented.

In the Office Action, prior pending claims 1 & 3-9 were rejected under 35 U.S.C. §102(b) as being anticipated by Nelson et al. (U.S. Patent No. 5,568,641; hereinafter Nelson), and claim 10 was rejected under 35 U.S.C. §103(a) as being unpatentable over Nelson. These rejections are respectfully traversed to any extent deemed applicable to the claims presented herewith, and reconsideration thereof is requested for the reasons set forth below.

Numerous aspects of Applicant's recited independent claim 1 are believed to patentably distinguish over the applied and known art. For example, Applicant recites copying, by a currently executing, first software module of a system, update control code from the currently executing, first software module to memory space outside a memory location at which the first software module is currently executing. As recited in amended claim 1, *only the update control code is copied from the currently executing, first software module to the memory space outside*

*the memory location at which the first software module is currently executing.* By copying only the update control code, only a small portion of the currently executing, first software module is copied, which has certain advantages. For example, as described at paragraph [0008] of the application, in some embedded systems, memory may be constrained, leaving no room for modules while the old module is still present. Additionally, by only copying the update control code, Applicant is able to achieve a faster update of the currently executing module.

In contrast, Nelson teaches that the entire boot block is copied to the alternate area. Nelson refers to manipulation within separately erasable/writable blocks of a flash EEPROM (column 2, line 15) or RAM made nonvolatile by means of power supply backup (column 3, line 30). Nelson teaches at page 7, column 3, line 22, that it requires the memory to have at least two separately erasable/writable blocks. The entire boot block is copied to this alternate area (column 2, line 25 of Nelson), not a subset of that code. Nelson actually requires that there be sufficient memory available to perform copying of the entire boot block. Applicant's invention is contrasted with this teaching, wherein only a small portion of the module, that is, the update control code, is copied from the currently executing, first software module to the memory space outside the memory location of which the first software module is currently executing.

In addition, Applicant's non-disruptive protocol of amended claim 1 recites that the executing of the update control code copied from the first software module monitors replacing of the first software module with the second software module. Cited against this aspect of Applicant's invention is column 3, lines 35-43 of Nelson. These lines teach:

Block 0 is the primary boot block for microprocessor (processor) system 20, and is addressable at a primary address space. Primary boot block means that processor 20 looks to the primary address range associated with block 0 upon power up or initialization for primary boot firmware execution. The primary boot block must contain the subset of firmware functionality required to bring the processor to an operational state sufficient to perform an upgrade.

Applicant respectfully submit that the above-noted teaching states that the primary boot block contains a subset of firmware functionality required to bring the processor to an operational state sufficient to perform an upgrade. This bringing of the processor to an operational state sufficient to perform an upgrade does not equate to Applicants' recited protocol of executing the update control code to monitor replacing of the first software module with the

second software module. No *monitoring* of the replacing or updating process is described at column 3, lines 35-43 of Nelson.

Still further, Applicant's amended claim 1 recites that, responsive to the replacing, *and without resetting or restarting the system*, branching from executing the update control code to executing code of the second software module at an entry point thereof specified by the update control code, thereby beginning execution of the second software module. There is no teaching or suggestion in Nelson of beginning execution of the second software module without resetting or restarting the system.

Column 5, line 3 of Nelson states that all blocks of memory are erased, other than the primary boot block. It also states that the boot firmware may need to be copied out of the primary boot block and into processor memory for execution. It is obvious that Nelson means that the code doing the update is either in processor memory (other than non-volatile memory) or it is the boot block 0 in the non-volatile memory. Either approach would work. Nelson describes how the non-volatile memory is updated, and that the *updating* of the non-volatile memory does not require a reset (disruptive). *However, Nelson does not address how this new code then becomes the operational code in the system.* One skilled in the art would assume that it requires a disruptive reset. If it doesn't, how are the processor instruction caches invalidated of the old instructions? Can it handle having a different entry point for boot block 2? Applicant's recited invention describes how this can be done in a non-disruptive manner. Further, Applicant respectfully submits that his recited protocol would not have been inherent in the teachings of Nelson. A careful reading of Nelson fails to uncover any line of reasoning that the recited functionality of Applicant's invention *necessarily* flows from the teachings of Nelson. In fact, the conventional approach to beginning execution of the second software module is to reset or restart the system.

Yet further, Applicant's amended claim 1 recites that the first software module and the second software module are each a single statically linked module. In contrast, Nelson describes the boot (at initialization) block many times, and at column 3, line 35, it describes block 0 as the initialization code for the system, and on lines 50-55, Nelson states that blocks 1, 2 & 3 are usually for the "other firmware necessary for the functional purposes of the processor". Thus,

Nelson teaches that the firmware of the system is broken up into many pieces, and is not a single, statically linked module, as recited in Applicant's independent claim 1. Further, Nelson teaches erasing different portions, and moving the portions around during an update. Nelson teaches how a disruption during the update of boot block 0 does not render the system useless, but it does not teach anything about the affects of having one piece of firmware at one level while another is at a different level. Presumably, there is firmware in boot block 0 to check for these conditions and somehow handle them. These problems are avoided in Applicant's invention by requiring that the module being replaced be a single statically linked module.

For at least the above-noted reasons, Applicant respectfully submits that independent claim 1 patentably distinguishes over Nelson, as well as the other art of record. Reconsideration and withdrawal of the rejection thereto is therefore respectfully requested.

The remaining dependent claims are believed allowable for the same reasons as independent claim 1, as well as for their own additional characterizations. For example, dependent claim 3 specifies that the first software module is an operating firmware module. This is distinct from the boot block or initialization code being updated in Nelson.

All claims are believed to be in condition for allowance, and such action is respectfully requested.

*Should any issue remain unresolved, however, Applicant's undersigned representative requests a telephone interview with the Examiner to discuss the pending claims in the hope of advancing prosecution of the subject application.*

Respectfully submitted,

  
\_\_\_\_\_  
Kevin P. Radigan  
Attorney for Applicant  
Registration No.: 31,789

Dated: November 19, 2008.

HESLIN ROTHENBERG FARLEY & MESITI P.C.  
5 Columbia Circle  
Albany, New York 12203-5160  
Telephone: (518) 452-5600  
Facsimile: (518) 452-5579